C 13

```
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSS
RRR   RRR        MMM         MMM               SSS
RRR   RRR        MMM         MMM               SSS
RRR   RRR        MMM         MMM               SSS
RRR      RRR     MMM         MMM               SSS
RRR      RRR     MMM         MMM               SSS
RRR      RRR     MMM         MMM               SSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSSS
```

```
**FILE**ID**RM3BKTSPL                                    I 13




RRRRRRRR    MM      MM    333333    BBBBBBBB    KK      KK  TTTTTTTTTT    SSSSSSSS  PPPPPPPP    LL
RRRRRRRR    MM      MM    333333    BBBBBBBB    KK      KK  TTTTTTTTTT    SSSSSSSS  PPPPPPPP    LL
RR      RR  MMMM  MMMM  33      33  BB      BB  KK      KK      TT      SS          PP      PP  LL
RR      RR  MMMM  MMMM  33      33  BB      BB  KK      KK      TT      SS          PP      PP  LL
RR      RR  MM  MM  MM          33  BB      BB  KK    KK        TT      SS          PP      PP  LL
RR      RR  MM  MM  MM          33  BB      BB  KK    KK        TT      SS          PP      PP  LL
RRRRRRRR    MM      MM        33    BBBBBBBB    KKKKKK          TT        SSSSSS    PPPPPPPP    LL
RRRRRRRR    MM      MM        33    BBBBBBBB    KKKKKK          TT        SSSSSS    PPPPPPPP    LL
RR  RR      MM      MM          33  BB      BB  KK    KK        TT              SS  PP          LL
RR  RR      MM      MM          33  BB      BB  KK    KK        TT              SS  PP          LL
RR    RR    MM      MM  33      33  BB      BB  KK      KK      TT              SS  PP          LL             ....
RR    RR    MM      MM  33      33  BB      BB  KK      KK      TT              SS  PP          LL             ....
RR      RR  MM      MM    333333    BBBBBBBB    KK      KK      TT      SSSSSSSS    PP          LLLLLLLLLL     ....
RR      RR  MM      MM    333333    BBBBBBBB    KK      KK      TT      SSSSSSSS    PP          LLLLLLLLLL     ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
0
0 MODULE RM3BKTSPL (LANGUAGE (BLISS32) ,
0                 IDENT = 'V04-000'
0                 ) =
1 BEGIN
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1
1 !++
1 !
1 ! FACILITY.       RMS32 INDEX SEQUENTIAL FILE ORGANIZATION
1 !
1 ! ABSTRACT:
1 !                 Routine to move out data in case of a split
1 !
1 !
1 ! ENVIRONMENT:
1 !
1 !                 VAX/VMS OPERATING SYSTEM
1 !
1 !--
1
1 !
1 ! AUTHOR:         Wendy Koenig             17-Jul-1978
1 !
1 ! MODIFIED BY:
1 !
1 !       V03-006 MCN0014         Maria del C. Nasr       22-Mar-1983
1 !               More linkages reorganization.
1 !
1 !       V03-005 MCN0013         Maria del C. Nasr       23-Feb-1983
1 !               Reorganize linkages.
1 !
1 !       V03-004 KBT0155         Keith B. Thompson       31-Aug-1982
1 !               Reorganize psects
1 !
```

```
1 !      V03-003 TMK0001          Todd M. Katz            02-Jul-1982
1 !              Implement the RMS cluster solution for next record positioning.
1 !              There is no longer any need begin the process of updating the
1 !              NRP list as part of a bucket split because there is no longer
1 !              any NRP list to update. Next record positioning context is now
1 !              kept locally in the IRAB.
1 !
1 !              In addition, the RFA of the new record is always stored in
1 !              IRB$L_PUTUP_VBN, and IRB$W_PUTUPD_ID. This is because the
1 !              current record context never changes because of a $PUT or
1 !              $DELETE.
1 !
1 !      V03-002 KBT0064          Keith B. Thompson       17-Jun-1982
1 !              Remove ref. to rm$sig_chars
1 !
1 !      V03-001 LJA0007          Laurie Anderson         25-Mar-1982
1 !              Change KBUFSZ to reference a macro when computing buffer
1 !              size and make IFB$B_KBUFSZ a word, now: IFB$W_KBUFSZ.
1 !
1 !      V02-014 KPL0001          Peter Lieberwirth       19-Aug-1981
1 !              Preserve NEW_BKT NXTRECID field as it was set up by
1 !              RM$ALLOC_BKT instead of resetting it to 1.  This
1 !              permits space reclamation to work by not reusing old
1 !              IDs in any new incarnations of the bucket.
1 !
1 !      V02-013 MCN0012          Maria del C. Nasr       07-Jul-1981
1 !              Recompress key of record which follows record inserted.
1 !              Also, fix some problems with 4-bucket splits and significant
1 !              characters.
1 !
1 !      V02-012 MCN0011          Maria del C. Nasr       26-May-1981
1 !              Add support for prologue 3 files.
1 !
1 !      V02-011 MCN0006          Maria del C. Nasr       16-Mar-1981
1 !              Increase size of record identifier to a word in NRP.
1 !
1 !      V02-010 REFORMAT         Frederick E. Deen, Jr.      23-Jul-1980
1 !              This code was reformatted to adhere to RMS standards
1 !
1 !
1 ! REVISION HISTORY:
1 !
1 !   Wendy Koenig,       21-Sep-1978
1 !   X0002 - Don't zero NRP list for each new bucket
1 !
1 !   Wendy Koenig,       25-Sep-1978
1 !   X0003 - Don't update RP on split -- it's an RRV
1 !
1 !   Christian Saether,  4-Oct-1978
1 !   X0004 - Modifications for UPDATE
1 !
1 !   Wendy Koenig,       12-Oct-1978
1 !   X0005 - Take all the NRP stuff out of here
1 !
1 !   Wendy Koenig,       19-Oct-1978
1 !   X0006 - Make some changes for the NEW_VBN entry in the NRP list
1 !
```

```
;  115          0115  1 !     Wendy Koenig,       24-Oct-1978
;  116          0116  1 !     X0007 - Make changes caused by sharing conventions
;  117          0117  1 !
;  118          0118  1 !     Christian Saether,  19-Dec-1978
;  119          0119  1 !     X0008 - Bliss does not like using AP as block structure
;  120          0120  1 !
;  121          0121  1 !     Wendy Koenig,       25-Jan-1979
;  122          0122  1 !     X0009 - Get rid of setting valid
;  123          0123  1 !
;  124          0124  1 !*****
;  125          0125  1
;  126          0126  1 LIBRARY 'RMSLIB:RMS';
;  127          0127  1
;  128          0128  1 REQUIRE 'RMSSRC:RMSIDXDEF';
;  129          0193  1
;  130          0194  1 ! define default psects for code
;  131          0195  1 !
;  132          0196  1
;  133          0197  1 PSECT
;  134          0198  1      CODE = RM$RMS3(PSECT_ATTR),
;  135          0199  1      PLIT = RM$RMS3(PSECT_ATTR);
;  136          0200  1
;  137          0201  1 ! Linkages
;  138          0202  1 !
;  139          0203  1
;  140          0204  1 LINKAGE
;  141          0205  1      L_JSB01,
;  142          0206  1      L_RABREG_4567,
;  143          0207  1      L_RABREG_67,
;  144          0208  1      L_REC_OVHD;
;  145          0209  1
;  146          0210  1 ! External Routines
;  147          0211  1 !
;  148          0212  1 EXTERNAL ROUTINE
;  149          0213  1      RM$BLDUDR            : RL$RABREG_4567,
;  150          0214  1      RM$EXPAND_KEY        : RL$JSB01,
;  151          0215  1      RM$GETNEXT_REC       : RL$RABREG_67,
;  152          0216  1      RM$REC_OVHD          : RL$REC_OVHD,
;  153          0217  1      RM$RECOMPR_KEY       : RL$JSB01;
;  154          0218  1
```

```
1 %SBTTL  'RM$BKT_SPL'
1 GLOBAL ROUTINE RM$BKT_SPL(RECSZ) : RL$RABREG_67 NOVALUE =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Move data records out a bucket that's splitting.
1 !
1 ! CALLING SEQUENCE:
1 !       BSBW RM$BKT_SPL()
1 !
1 ! INPUT PARAMETERS:
1 !
1 !       RECSZ - record size of record to be inserted
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       IRAB    SPLIT, SPLIT_1, SPLIT_2, POS_INS,
1 !               NEW_BKTS, BKT_NO, REC_W_LO,
1 !               CURBDB -- ORIGINAL BUCKET, NXTBDB -- NEW BUCKET
1 !       IN NEW BUCKET, NXTRECID
1 !       IFAB -- prologue version number
1 !       RAB for RSZ, RBF
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       BKT_NO is decremented
1 !       FREESPACE and NXTID in new bkt is set
1 !
1 ! ROUTINE VALUE:
1 !       nothing
1 !
1 ! SIDE EFFECTS:
1 !       Data records are moved from one bucket to another.
1 !       The records are assigned new ids, in numerical order.
1 !       The RFA address of current record becomes the RFA address of the new
1 !         record if the new record was inserted into the new bucket.
1 !       Mark new bucket dirty and valid.
1 !       If the primary key is compressed, the key in the first record of the
1 !         new bucket undergoes expansion.
1 !       AP is clobbered.
1 !
1 !--
1
2     BEGIN
2
2     EXTERNAL REGISTER
2         R_REC_ADDR_STR,
2         R_IDX_DFN_STR,
2         R_IFAB_STR,
2         R_IRAB_STR,
2         R_RAB_STR;
2
2     GLOBAL REGISTER
```

```
        R_IMPURE;

    LOCAL
        NEW_BKT : REF BBLOCK,
        OLD_BKT : REF BBLOCK,
        NEXT_REC: REF BBLOCK,
        EOB,
        SPLIT_PT        : WORD,
        FLAG    : BLOCK [1];

    BUILTIN
        AP;

    MACRO
        NEW_VBN = 0,0,2,0 %,
        ALONE = 0,2,1,0 %;

    BUILTIN
        TESTBITCC;

    ! Set up NEW_BKT and OLD_BKT addresses.
    !
    NEW_BKT = .BBLOCK[.IRAB[IRB$L_NXTBDB], BDB$L_ADDR];
    OLD_BKT = .BBLOCK[.IRAB[IRB$L_CURBDB], BDB$L_ADDR];

    ! Set up SPLIT_PT and EOB for this move. Also set up AP to signal if the new
    ! record belongs by itself. If this is the only new bucket, the new record
    ! may be positioned at the end of the new bucket w/o REC_W_LO being set.
    ! Therefore we can set it.
    !
    FLAG = 1;                       ! one indicates VBN_RIGHT ( " the default")

    CASE .IRAB[IRB$V_BKT_NO] FROM 1 TO 3 OF
        SET

        [3] :

            BEGIN
            SPLIT_PT = .IRAB[IRB$W_SPLIT_2];
            REC_ADDR = .OLD_BKT + BKT$C_OVERHDSZ;
            EOB = .OLD_BKT + .OLD_BKT[BKT$W_FREESPACE];

            DO
                BEGIN

                IF .REC_ADDR[IRC$V_RRV]
                THEN
                    EXITLOOP;

                RM$GETNEXT_REC()
                END
            UNTIL .REC_ADDR GEQU .EOB;

            EOB = .REC_ADDR - .OLD_BKT;
            END;
        [2] :
            BEGIN
```

```
;   270     0333  3                 SPLIT_PT = .IRAB[IRB$W_SPLIT_1];
;   271     0334  3                 EOB = .IRAB[IRB$W_SPLIT_2];
;   272     0335  3
;   273     0336  4                 BEGIN
;   274     0337  4
;   275     0338  4                     IF .SPLIT_PT EQLU .IRAB[IRB$W_POS_INS]
;   276     0339  4                     AND
;   277     0340  4                     .SPLIT_PT EQLU .IRAB[IRB$W_SPLIT]
;   278     0341  4                     THEN
;   279     0342  4                         FLAG[ALONE] = 1;
;   280     0343  3                 END;
;   281     0344  3
;   282     0345  3             IF .IRAB[IRB$L_VBN_MID] NEQ 0
;   283     0346  3             THEN
;   284     0347  3                 FLAG[NEW_VBN] = 3;
;   285     0348  2             END;
;   286     0349  2
;   287     0350  2         [1] :
;   288     0351  2
;   289     0352  3             BEGIN
;   290     0353  3             SPLIT_PT = .IRAB[IRB$W_SPLIT];
;   291     0354  3             EOB = .IRAB[IRB$W_SPLIT_1];
;   292     0355  3
;   293     0356  3             IF .IRAB[IRB$L_VBN_MID] NEQ 0
;   294     0357  3             THEN
;   295     0358  3                 FLAG[NEW_VBN] = 2;
;   296     0359  3
;   297     0360  4             IF (.EOB<0, 16> EQLU .IRAB[IRB$W_POS_INS])
;   298     0361  3                 AND
;   299     0362  4                 (.SPLIT_PT NEQU .EOB<0, 16>)
;   300     0363  3                 AND
;   301     0364  4                 ( NOT .IRAB[IRB$V_BIG_SPLIT])
;   302     0365  3             THEN
;   303     0366  3                 IRAB[IRB$V_REC_W_LO] = 1;
;   304     0367  2             END;
;   305     0368  2
;   306     0369  2         TES;
;   307     0370  2
;   308     0371  2     ! If the new record belongs in the middle of the new bucket, we have to do
;   309     0372  2     ! the move in three pieces; 1) Move out the "hi set", 2) build record in
;   310     0373  2     ! the new bucket, and 3) move out "lo set". Note that the hi set and / or
;   311     0374  2     ! lo set may be non-existent.
;   312     0375  2     !
;   313     0376  2     NEXT_REC = 0;                           ! assume record does not go in this bucket
;   314     0377  2
;   315     0378  2     IF .SPLIT_PT LEQU .IRAB[IRB$W_POS_INS]
;   316     0379  2       AND
;   317     0380  2       .IRAB[IRB$W_POS_INS] LEQU .EOB<0, 16>
;   318     0381  2     THEN
;   319     0382  3         BEGIN
;   320     0383  3         REC_ADDR = CH$MOVE(.IRAB[IRB$W_POS_INS] - .SPLIT_PT,
;   321     0384  3             .SPLIT_PT + .OLD_BKT, .NEW_BKT + BKT$C_OVERHDSZ);
;   322     0385  4         BEGIN
;   323     0386  4
;   324     0387  4         LABEL
;   325     0388  4             BUILD;
;   326     0389  4
```

```
RM3
Sym

$$.
$$R
$$R
$$R
$$R
BUG
RM$
SYS

PSE
---
.
RM$

Pha
---
Ini
Com
Pas
Sym
Pas
Sym
Pse
Cro
Ass

The
145
The
104
5 p

Mac
---
_$2
_$2
_$2
TOT

32

The

MAC
```

```
;    327          0390  4            GLOBAL REGISTER
;    328          0391  4                COMMON_IOREG;
;    329          0392  4
;    330          0393  4            BKT_ADDR = .NEW_BKT;
;    331          0394  4            BDB = .IRAB[IRB$L_NXTBDB];
;    332          0395  4  BUILD :
;    333          0396  4
;    334          0397  4            ! If so desired, now is the time to build the user data record in the
;    335          0398  4            ! new bkt.  The ID for this record will be zeroed, and filled when
;    336          0399  4            ! the record ID's for the other records are reassigned.
;    337          0400  4            !
;    338          0401  5            BEGIN
;    339          0402  5
;    340          0403  5            IF .SPLIT_PT EQLU .IRAB[IRB$W_POS_INS]
;    341          0404  5            THEN
;    342          0405  6                BEGIN
;    343          0406  6
;    344          0407  6                IF NOT .IRAB[IRB$V_REC_W_LO]
;    345          0408  6                    AND
;    346          0409  6                    NOT .FLAG[ALONE]
;    347          0410  6                THEN
;    348          0411  7                    BEGIN
;    349          0412  7                    NEXT_REC = 1;
;    350          0413  7                    RM$B[DUDR(.RECSZ);
;    351          0414  6                    END;
;    352          0415  6
;    353          0416  6                LEAVE BUILD
;    354          0417  6
;    355          0418  5                END;
;    356          0419  5
;    357          0420  5            IF .EOB<0, 16> EQLU .IRAB[IRB$W_POS_INS]
;    358          0421  5            THEN
;    359          0422  6                BEGIN
;    360          0423  6
;    361          0424  6                IF .IRAB[IRB$V_REC_W_LO]
;    362          0425  6                THEN
;    363          0426  7                    BEGIN
;    364          0427  7                    NEXT_REC = 1;
;    365          0428  7                    RM$B[DUDR(.RECSZ);
;    366          0429  6                    END;
;    367          0430  6
;    368          0431  6                LEAVE BUILD;
;    369          0432  6
;    370          0433  5                END;
;    371          0434  5
;    372          0435  5            ! At this point the only case is that POS_INS is in the middle of the
;    373          0436  5            ! bucket so we always want to insert the new record.
;    374          0437  5            !
;    375          0438  5            NEXT_REC = 1;
;    376          0439  5            RM$B[DUDR(.RECSZ);
;    377          0440  4            END;                                        ! {end of build }
;    378          0441  3            END;
;    379          0442  3
;    380          0443  3            ! If the record was written to this bucket, and there will be a hi set
;    381          0444  3            ! to move, then set the flag to the address of the record after the one
;    382          0445  3            ! inserted.  Otherwise, clear indicator.
;    383          0446  3            !
```

```
3
3           IF .NEXT_REC
3             AND (.EOB<0,16> - .IRAB[IRB$W_POS_INS]) NEQU 0
3           THEN
3               NEXT_REC = .REC_ADDR
3           ELSE
3               NEXT_REC = 0;
3           REC_ADDR = CH$MOVE(.EOB<0, 16> - .IRAB[IRB$W_POS_INS],
3                                  .IRAB[IRB$W_POS_INS] + .OLD_BKT,
3                                  .REC_ADDR);
3           END
2       ELSE
2
2             ! The new record does not go into new bucket so just move data out in
2             ! one chunk.
2             !
2           REC_ADDR = CH$MOVE(.EOB<0, 16> - .SPLIT_PT,
2                                  .SPLIT_PT + .OLD_BKT,
2                                  .NEW_BKT + BKT$C_OVERHDSZ);
2
2       ! Re-allocate the ID's, in numerical order, for the new bucket. While RMS
2       ! is doing this it assigns the ID to the new record, if the new record
2       ! goes in the new bucket.
2       !
3       BEGIN
3
3       EOB = .REC_ADDR;
3
3       ! If the record was inserted into this bucket, BLDUDR incremented NXTRECID.
3       ! Renumber the IDs in the new bucket.  Do it differently, depending on
3       ! prologue version number.
3       !
3       REC_ADDR = .NEW_BKT + BKT$C_OVERHDSZ;
3
3       IF .IFAB[IFB$B_PLG_VER] LSSU PLG$C_VER_3
3       THEN
3           WHILE .REC_ADDR LSSU .EOB
3           DO
4               BEGIN
4
4                 ! If the ID of the record RMS is currently positioned to is 0,
4                 ! then it is the new record. In this case, the ID of the RRV also
4                 ! has to be set as well as the ID field of the RFA address of the
4                 ! next record positioning context's current record.
4                 !
4               IF .REC_ADDR[IRC$B_ID] EQLU 0
4               THEN
5                   BEGIN
5                   (.REC_ADDR + IRC$C_DATOVHDSZ)<0, 8> = .NEW_BKT[BKT$B_NXTRECID];
5                   IRAB[IRB$W_PUTUP_ID] = .NEW_BKT[BKT$B_NXTRECID];
4                   END;
4
4               REC_ADDR[IRC$B_ID] = .NEW_BKT[BKT$B_NXTRECID];
4               NEW_BKT[BKT$B_NXTRECID] = .NEW_BKT[BKT$B_NXTRECID] + 1;
4               RM$GETNEXT_REC()
4               END                                 ! end of while loop
3       ELSE
```

```
;   441         0504  3          WHILE .REC_ADDR LSSU .EOB
;   442         0505  3          DO
;   443         0506  4              BEGIN
;   444         0507  4
;   445         0508  4              ! If the ID of the record RMS is currently positioned to is 0,
;   446         0509  4              ! then it is the new record. In this case, the ID of the RRV also
;   447         0510  4              ! has to be set as well as the ID field of the RFA address of the
;   448         0511  4              ! next record positioning context's current record.
;   449         0512  4              !
;   450         0513  4              IF .REC_ADDR[IRC$W_ID] EQLU 0
;   451         0514  4              THEN
;   452         0515  5                  BEGIN
;   453         0516  5                  (.REC_ADDR + IRC$C_DATOVHSZ3)<0,16> = .NEW_BKT[BKT$W_NXTRECID];
;   454         0517  5                  IRAB[IRB$W_PUTUP_ID] = .NEW_BKT[BKT$W_NXTRECID];
;   455         0518  4                  END;
;   456         0519  4
;   457         0520  4              REC_ADDR[IRC$W_ID] = .NEW_BKT[BKT$W_NXTRECID];
;   458         0521  4              NEW_BKT[BKT$W_NXTRECID] = .NEW_BKT[BKT$W_NXTRECID] + 1;
;   459         0522  4              RM$GETNEXT_REC()
;   460         0523  3              END;                            ! end of while loop
;   461         0524  3
;   462         0525  2      END:                                    ! { end of block redefining eob }
;   463         0526  2
;   464         0527  2      BBLOCK[.IRAB[IRB$L_NXTBDB], BDB$V_DRT] = 1;
;   465         0528  2      NEW_BKT[BKT$W_FREESPACE] = .REC_ADDR - .NEW_BKT;
;   466         0529  2
;   467         0530  2      ! If the record was inserted in this bucket followed by another record
;   468         0531  2      ! which is not an RRV, and the key is compressed, then recompress the key
;   469         0532  2      ! of the record which follows the inserted record.
;   470         0533  2      !
;   471         0534  2
;   472         0535  2      IF .NEXT_REC NEQU 0
;   473         0536  2        AND .IDX_DFN[IDX$V_KEY_COMPR]
;   474         0537  2      THEN
;   475         0538  2
;   476         0539  2          IF NOT .NEXT_REC[IRC$V_RRV]
;   477         0540  2          THEN
;   478         0541  3              BEGIN
;   479         0542  3
;   480         0543  3              GLOBAL REGISTER
;   481         0544  3                  R_BKT_ADDR;
;   482         0545  3
;   483         0546  3              LOCAL
;   484         0547  3                  TMP_REC_ADDR;
;   485         0548  3
;   486         0549  3              BKT_ADDR = .NEW_BKT;
;   487         0550  3              TMP_REC_ADDR = .REC_ADDR;
;   488         0551  3              REC_ADDR = .NEXT_REC;
;   489         0552  3              RM$RECOMPR_KEY ( .IRAB[IRB$L_RECBUF],
;   490         0553  3                                .REC_ADDR + RM$REC_OVHD(0) );
;   491         0554  3              REC_ADDR = .TMP_REC_ADDR;
;   492         0555  2              END;
;   493         0556  2
;   494         0557  3      BEGIN
;   495         0558  3
;   496         0559  3      LOCAL
;   497         0560  3          SIG_FLG,
;  498          0561  3             KEY_ADDR1,
;  499          0562  3             KEY_ADDR2;
;  500          0563  3
;  501          0564  3         ! Determine which key buffer contains the last key of the previous bucket.
;  502          0565  3         ! If we are allocating bucket 2 or 3 of a big split, then keybuffer3 (and
;  503          0566  3         ! keybuffer5) contains the key.  Otherwise, it is in keybuffer2.
;  504          0567  3         !
;  505          0568  3
;  506          0569  3         IF .IRAB[IRB$V_BKT_NO] GTRU 1
;  507          0570  3         THEN
;  508          0571  4             BEGIN
;  509          0572  4             SIG_FLG = 0;
;  510          0573  4             KEY_ADDR1 = KEYBUF_ADDR(5);
;  511          0574  4             KEY_ADDR2 = KEYBUF_ADDR(3);
;  512          0575  4             END
;  513          0576  3         ELSE
;  514          0577  4             BEGIN
;  515          0578  4             SIG_FLG = 2;
;  516          0579  4             KEY_ADDR1 = KEY_ADDR2 = KEYBUF_ADDR(2);
;  517          0580  3             END;
;  518          0581  3
;  519          0582  3         ! If the primary key is compressed, we must expand the first key of the
;  520          0583  3         ! new bucket, since it cannot be front end compressed.  Base this expansion
;  521          0584  3         ! on what will be the last key of the previous bucket, obtained from the
;  522          0585  3         ! right key buffer.
;  523          0586  3         !
;  524          0587  3         IF .IDX_DFN[IDX$V_KEY_COMPR]
;  525          0588  3         THEN
;  526          0589  4             BEGIN
;  527          0590  4
;  528          0591  4             GLOBAL REGISTER
;  529          0592  4                 R_BKT_ADDR;
;  530          0593  4
;  531          0594  4             RM$EXPAND_KEY ( .KEY_ADDR1, .NEW_BKT );
;  532          0595  3             END;
;  533          0596  3
;  534          0597  2         END;                    ! end of local definition for KEY_ADDR
;  535          0598  2
;  536          0599  2         ! Since I know that BKT_NO is a 2-bit digit ranging from 1 to 3, I can
;  537          0600  2         ! optimize the decr desired, so bear with me. Note: BKT_NO_LO refers to
;  538          0601  2         ! the low bit of BKT_NO.
;  539          0602  2         !
;  540          0603  2         IF TESTBITCC(IRAB[IRB$V_BKT_NO_LO])
;  541          0604  2         THEN
;  542          0605  2             IRAB[IRB$V_BKT_NO] = 1;
;  543          0606  2
;  544          0607  2         RETURN;
;  545          0608  2
;  546          0609  1         END;                                                    ! { end of rm$bkt_spl }


                                                                                         .TITLE  RM3BKTSPL
                                                                                         .IDENT  \V04-000\

                                                                                         .EXTRN  RM$BLDUDR, RM$EXPAND_KEY
                                                                                         .EXTRN  RM$GETNEXT_REC, RM$REC_OVHD
                                                                                         .EXTRN  RM$RECOMPR_KEY
```

```
                                                                        G 14
RM3BKTSPL                                                              16-Sep-1984 01:37:40     VAX-11 Bliss-32 V4.0-742                  Page  11
V04-000          RM$BKT_SPL                                            14-Sep-1984 13:01:14     [RMS.SRC]RM3BKTSPL.B32;1                       (2)

                                                                                  .PSECT   RM$RMS3,NOWRT,  GBL,  PIC,2

                                                    083C   8F  BB 00000 RM$BKT_SPL::
                                                                                  PUSHR    #^M<R2,R3,R4,R5,R11>                             ; 0220
                                            5E             18  C2 00004           SUBL2    #24, SP
                                            50        3C   A9  D0 00007           MOVL     60(IRAB), R0                                     ; 0298
                                      0C    AE        18   A0  D0 0000B           MOVL     24(R0), NEW_BKT
                                            50        20   A9  D0 00010           MOVL     32(IRAB), R0                                     ; 0299
                                      08    AE        18   A0  D0 00014           MOVL     24(R0), OLD_BKT
                                      04    AE             01  D0 00019           MOVL     #1, FLAG                                         ; 0306
             52        44    A9             02             00  EF 0001D           EXTZV    #0, #2, 68(IRAB), R2                             ; 0308
                             02             01             52  CF 00023           CASEL    R2, #1, #2
                           0006           0032           0057     00027 1$:       .WORD    7$-1$,-
                                                                                           5$-1$,-
                                                                                           2$-1$
                                            6E        4E   A9  B0 0002D 2$:       MOVW     78(IRAB), SPLIT_PT                               ; 0314
                             56       08    AE             0E  C1 00031           ADDL3    #14, OLD_BKT, REC_ADDR                           ; 0315
                             51       08    AE             04  C1 00036           ADDL3    #4, OLD_BKT, R1                                  ; 0316
                                            50             61  3C 0003B           MOVZWL   (R1), R0
                                      10    AE        08 BE40  9E 0003E           MOVAB    @OLD_BKT[R0], EOB
                             09             66             03  E0 00044 3$:       BBS      #3, (REC_ADDR), 4$                               ; 0321
                                                        0000G  30 00048           BSBW     RM$GETNEXT_REC                                   ; 0325
                                      10    AE             56  D1 0004B           CMPL     REC_ADDR, EOB                                    ; 0327
                                                           F3  1F 0004F           BLSSU    3$
                       10    AE             56        08   AE  C3 00051 4$:       SUBL3    OLD_BKT, REC_ADDR, EOB                           ; 0329
                                                           50  11 00057           BRB      9$                                               ; 0308
                                            6E        4C   A9  B0 00059 5$:       MOVW     76(IRAB), SPLIT_PT                               ; 0333
                                      10    AE        4E   A9  3C 0005D           MOVZWL   78(IRAB), EOB                                    ; 0334
                                      48    A9             6E  B1 00062           CMPW     SPLIT_PT, 72(IRAB)                               ; 0338
                                                           0A  12 00066           BNEQ     6$
                                      4A    A9             6E  B1 00068           CMPW     SPLIT_PT, 74(IRAB)                               ; 0340
                                                           04  12 0006C           BNEQ     6$
                                      04    AE             04  88 0006E           BISB2    #4, FLAG                                         ; 0342
                                                    0090   C9  D5 00072 6$:       TSTL     144(IRAB)                                        ; 0345
                                                           31  13 00076           BEQL     9$
                                      04    AE             03  88 00078           BISB2    #3, FLAG                                         ; 0347
                                                           2B  11 0007C           BRB      9$                                               ; 0308
                                            6E        4A   A9  B0 0007E 7$:       MOVW     74(IRAB), SPLIT_PT                               ; 0353
                                      10    AE        4C   A9  3C 00082           MOVZWL   76(IRAB), EOB                                    ; 0354
                                                    0090   C9  D5 00087           TSTL     144(IRAB)                                        ; 0356
                                                           06  13 0008B           BEQL     8$
       04    AE              02             00             02  F0 0008D           INSV     #2, #0, #2, FLAG                                 ; 0358
                                      48    A9        10   AE  B1 00093 8$:       CMPW     EOB, 72(IRAB)                                    ; 0360
                                                           0F  12 00098           BNEQ     9$
                                      10    AE             6E  B1 0009A           CMPW     SPLIT_PT, EOB                                    ; 0362
                                                           09  13 0009E           BEQL     9$
                             04       44    A9             02  E0 000A0           BBS      #2, 68(IRAB), 9$                                 ; 0364
                                      44    A9             08  88 000A5           BISB2    #8, 68(IRAB)                                     ; 0366
                                                      14   AE  D4 000A9 9$:       CLRL     NEXT_REC                                         ; 0376
                                            50             6E  3C 000AC           MOVZWL   SPLIT_PT, R0                                     ; 0384
                                      48    A9             6E  B1 000AF           CMPW     SPLIT_PT, 72(IRAB)                               ; 0378
                                                           79  1A 000B3           BGTRU    15$
                                      10    AE        48   A9  B1 000B5           CMPW     72(IRAB), EOB                                    ; 0380
                                                           72  1A 000BA           BGTRU    15$
                                            51        48   A9  3C 000BC           MOVZWL   72(IRAB), R1                                     ; 0383
                                            52             6E  3C 000C0           MOVZWL   SPLIT_PT, R2
```

```
                   51             52  C2 000C3          SUBL2   R2, R1                                  ;
      7E     0C    AE             0E  C1 000C6          ADDL3   #14, NEW_BKT, -(SP)                     ; 0384
      9E     0C  BE40             51  28 000CB          MOVC3   R1, @OLD_BKT[R0], @(SP)+                ;
                   56             53  D0 000D1          MOVL    R3, REC_ADDR                            ;
                   55       0C    AE  D0 000D4          MOVL    NEW_BKT, BKT_ADDR                       ; 0393
                   54       3C    A9  D0 000D8          MOVL    60(IRAB), BDB                           ; 0394
             48    A9             6E  B1 000DC          CMPW    SPLIT_PT, 72(IRAB)                      ; 0403
                                  0C  12 000E0          BNEQ    10$                                     ;
      20     44    A9             03  E0 000E2          BBS     #3, 68(IRAB), 12$                       ; 0407
      1B     04    AE             02  E0 000E7          BBS     #2, FLAG, 12$                           ; 0409
                                  0C  11 000EC          BRB     11$                                     ; 0412
             48    A9       10    AE  B1 000EE 10$:     CMPW    EOB, 72(IRAB)                           ; 0420
                                  05  12 000F3          BNEQ    11$                                     ;
      0D     44    A9             03  E1 000F5          BBC     #3, 68(IRAB), 12$                       ; 0424
             14    AE             01  D0 000FA 11$:     MOVL    #1, NEXT_REC                            ; 0438
                            30    AE  DD 000FE          PUSHL   RECSZ                                   ; 0439
                               0000G  30 00101          BSBW    RM$BLDUDR                               ;
                   5E             04  C0 00104          ADDL2   #4, SP                                  ;
                   0D       14    AE  E9 00107 12$:     BLBC    NEXT_REC, 13$                           ; 0448
             48    A9       10    AE  B1 0010B          CMPW    EOB, 72(IRAB)                           ; 0449
                                  06  13 00110          BEQL    13$                                     ;
             14    AE             56  D0 00112          MOVL    REC_ADDR, NEXT_REC                      ; 0451
                                  03  11 00116          BRB     14$                                     ;
                            14    AE  D4 00118 13$:     CLRL    NEXT_REC                                ; 0453
                   50       48    A9  3C 0011B 14$:     MOVZWL  72(IRAB), R0                            ; 0454
                   51       10    AE  3C 0011F          MOVZWL  EOB, R1                                 ;
                   51             50  C2 00123          SUBL2   R0, R1                                  ;
      66     08  BE40             51  28 00126          MOVC3   R1, @OLD_BKT[R0], (REC_ADDR)            ; 0456
                                  15  11 0012C          BRB     16$                                     ;
                   51       10    AE  3C 0012E 15$:     MOVZWL  EOB, R1                                 ; 0463
                   52             6E  3C 00132          MOVZWL  SPLIT_PT, R2                            ;
                   51             52  C2 00135          SUBL2   R2, R1                                  ;
      7E     0C    AE             0E  C1 00138          ADDL3   #14, NEW_BKT, -(SP)                     ; 0465
      9E     0C  BE40             51  28 0013D          MOVC3   R1, @OLD_BKT[R0], @(SP)+                ;
                   56             53  D0 00143 16$:     MOVL    R3, REC_ADDR                            ;
             10    AE             56  D0 00146          MOVL    REC_ADDR, EOB                           ; 0473
      50     0C    AE             0E  C1 0014A          ADDL3   #14, NEW_BKT, R0                        ; 0479
                   56             60  9E 0014F          MOVAB   (R0), REC_ADDR                          ;
      50     0C    AE             06  C1 00152          ADDL3   #6, NEW_BKT, R0                         ; 0499
                   52             60  9E 00157          MOVAB   (R0), R2                                ;
                   03     00B7    CA  91 0015A          CMPB    183(IFAB), #3                           ; 0481
                                  29  1E 0015F          BGEQU   19$                                     ;
             10    AE             56  D1 00161 17$:     CMPL    REC_ADDR, EOB                           ; 0483
                                  4C  1E 00165          BGEQU   21$                                     ;
                            01    A6  95 00167          TSTB    1(REC_ADDR)                             ; 0492
                                  13  12 0016A          BNEQ    18$                                     ;
      50     0C    AE             06  C1 0016C          ADDL3   #6, NEW_BKT, R0                         ; 0495
             02    A6             60  90 00171          MOVB    (R0), 2(REC_ADDR)                       ;
      50     0C    AE             06  C1 00175          ADDL3   #6, NEW_BKT, R0                         ; 0496
           0080    C9             60  9B 0017A          MOVZBW  (R0), 128(IRAB)                         ;
             01    A6             62  90 0017F 18$:     MOVB    (R2), 1(REC_ADDR)                       ; 0499
                                  62  96 00183          INCB    (R2)                                    ; 0500
                               0000G  30 00185          BSBW    RM$GETNEXT_REC                          ; 0501
                                  D7  11 00188          BRB     17$                                     ;
             10    AE             56  D1 0018A 19$:     CMPL    REC_ADDR, EOB                           ; 0504
                                  23  1E 0018E          BGEQU   21$                                     ;
                            01    A6  B5 00190          TSTW    1(REC_ADDR)                             ; 0513
```

```
                                                         13 12 00193          BNEQ    20$                                             ;
                           50       0C    AE             06 C1 00195          ADDL3   #6, NEW_BKT, R0                                 ; 0516
                                    03    A6             60 B0 0019A          MOVW    (R0), 3(REC_ADDR)                               ;
                           50       0C    AE             06 C1 0019E          ADDL3   #6, NEW_BKT, R0                                 ; 0517
                                  0080    C9             60 B0 001A3          MOVW    (R0), 128(IRAB)                                 ;
                                    01    A6             62 B0 001A8 20$:     MOVW    (R2), 1(REC_ADDR)                               ; 0520
                                                         62 B6 001AC          INCW    (R2)                                            ; 0521
                                                      0000G 30 001AE          BSBW    RM$GETNEXT_REC                                  ; 0522
                                                         D7 11 001B1          BRB     19$                                             ;
                                          50       3C    A9 D0 001B3 21$:     MOVL    60(IRAB), R0                                    ; 0527
                                    0A    A0             02 88 001B7          BISB2   #2, 10(R0)                                      ;
                           50       0C    AE             04 C1 001BB          ADDL3   #4, NEW_BKT, R0                                 ; 0528
                           60             56       0C    AE A3 001C0          SUBW3   NEW_BKT, REC_ADDR, (R0)                         ;
                                                   14    AE D5 001C5          TSTL    NEXT_REC                                        ; 0535
                                                         28 13 001C8          BEQL    22$                                             ;
                           23       1C    A7             06 E1 001CA          BBC     #6, 28(IDX_DFN), 22$                            ; 0536
                           1E       14    BE             03 E0 001CF          BBS     #3, @NEXT_REC, 22$                              ; 0539
                                          55       0C    AE D0 001D4          MOVL    NEW_BKT, BKT_ADDR                               ; 0549
                                          52             56 D0 001D8          MOVL    REC_ADDR, TMP_REC_ADDR                          ; 0550
                                          56       14    AE D0 001DB          MOVL    NEXT_REC, REC_ADDR                              ; 0551
                                                         51 D4 001DF          CLRL    R1                                              ; 0553
                                                      0000G 30 001E1          BSBW    RM$REC_OVHD                                     ;
                           51             56             50 C1 001E4          ADDL3   R0, REC_ADDR, R1                                ;
                                          50       68    A9 D0 001E8          MOVL    104(IRAB), R0                                   ; 0552
                                                      0000G 30 001EC          BSBW    RM$RECOMPR_KEY                                  ;
                                          56             52 D0 001EF          MOVL    TMP_REC_ADDR, REC_ADDR                          ; 0554
                                          52     00B4    CA 9E 001F2 22$:     MOVAB   180(IFAB), R2                                   ; 0573
            01        44   A9             02             00 ED 001F7          CMPZV   #0, #2, 68(IRAB), #1                            ; 0569
                                                         11 1B 001FD          BLEQU   23$                                             ;
                                                         50 D4 001FF          CLRL    SIG_FLG                                         ; 0572
                                          51             62 3C 00201          MOVZWL  (R2), R1                                        ; 0573
                                          50       60  B941 DE 00204          MOVAL   @96(IRAB)[R1], KEY_ADDR1                        ;
                                          51       60  B941 3E 00209          MOVAW   @96(IRAB)[R1], KEY_ADDR2                        ; 0574
                                                         0D 11 0020E          BRB     24$                                             ; 0569
                                          50             02 D0 00210 23$:     MOVL    #2, SIG_FLG                                     ; 0578
                                          51             62 3C 00213          MOVZWL  (R2), KEY_ADDR2                                 ; 0579
                                          51       60    A9 C0 00216          ADDL2   96(IRAB), KEY_ADDR2                             ;
                                          50             51 D0 0021A          MOVL    KEY_ADDR2, KEY_ADDR1                            ;
                           07       1C    A7             06 E1 0021D 24$:     BBC     #6, 28(IDX_DFN), 25$                            ; 0587
                                          51       0C    AE D0 00222          MOVL    NEW_BKT, R1                                     ; 0594
                                                      0000G 30 00226          BSBW    RM$EXPAND_KEY                                   ;
                           06       44    A9             00 E4 00229 25$:     BBSC    #0, 68(IRAB), 26$                               ; 0603
      44    A9             02             00             01 F0 0022E          INSV    #1, #0, #2, 68(IRAB)                            ; 0605
                                          5E             18 C0 00234 26$:     ADDL2   #24, SP                                         ; 0609
                                                 083C    8F BA 00237          POPR    #^M<R2,R3,R4,R5,R11>                            ;
                                                            05 0023B          RSB                                                     ;

; Routine Size:   572 bytes,     Routine Base:   RM$RMS3 + 0000


;  547           0610   1
;  548           0611   1 END
;  549           0612   1
;  550           0613   0 ELUDOM
;                                      PSECT SUMMARY
;
;       Name                        Bytes                             Attributes
;
;   RM$RMS3                             572  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  GBL,  REL,  CON,  PIC,ALIGN(2)




;                               Library Statistics
;
;                                               -------- Symbols --------        Pages        Processing
;       File                                    Total    Loaded   Percent       Mapped       Time
;
;   _$255$DUA28:[RMS.OBJ]RMS.L32;1               3109        51         1        154           00:00.4






;                                      COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:RM3BKTSPL/OBJ=OBJ$:RM3BKTSPL MSRC$:RM3BKTSPL/UPDATE=(ENH$:RM3BKTSPL)

; Size:           572 code + 0 data bytes
; Run Time:          00:13.5
; Elapsed Time:      00:37.6
; Lines/CPU Min:       2722
; Lexemes/CPU-Min: 16410
; Memory Used:  214 pages
; Compilation Complete
```

0323 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY